Sensor-Based Change Detection for Timely Solicitation of User Engagement

Sensor-Based Change Detection for Timely Solicitation of User Engagement

Timothy Patterson, Naveed Khan, *Member, IEEE,* Sally McClean, *Member, IEEE,* Chris Nugent, *Member, IEEE,* Shuai Zhang, Ian Cleland, Qin Ni

***Abstract*—The accurate detection of changes has the potential to form a fundamental component of systems which autonomously solicit user interaction based on transitions within an input stream for example, electrocardiogram data or accelerometry obtained from a mobile device. This solicited interaction may be utilised for diverse scenarios such as responding to changes in a patient's vital signs within a medical domain or requesting user activity labels for generating real-world labelled datasets. Within this paper we extend our previous work on the Multivariate Online Change detection Algorithm subsequently exploring the utility of incorporating the Benjamini Hochberg method of correcting for multiple comparisons. Furthermore we evaluate our approach against similarly light-weight Multivariate Exponentially Weighted Moving Average and Cumulative Sum based techniques. Results are presented based on manually labelled change points in accelerometry data captured using 10 participants. Each participant performed 9 distinct activities for a total period of 35 minutes. The results subsequently demonstrate the practical potential of our approach from both accuracy and computational perspectives.**

***Index Terms*—Multivariate change detection, Online change detection, Soliciting user interaction**

I. Introduction

Many practical problems that are prevalent in areas such as fault detection and recognition-oriented signal processing can be modelled parametrically [1]. In these problem areas change detection refers to the identification of time points in which the parameters of a model are subject to sudden changes in characteristics at previously unknown time instants [1]. Change detection algorithms have been applied to diverse application domains such as detecting faults to ensure quality in a manufacturing process [2], detecting changes in seismic data to autonomously segment signals [3] and detecting changes in a patient's vital signs to alert medical personnel [4]. A further, emerging application domain for change point detection is autonomously identifying transitions in human activity, for example from 'stand' to 'walk'. Upon identification these transitions can then be utilised as the starting location for window-based activity recognition [5] and solicitation of manually labelled activity data [6].

T. Patterson, C. Nugent, S. Zhang and I. Cleland are with the School of Computing and Mathematics, Ulster University, Shore Road, Jordanstown, Newtownabbey, Co. Antrim, BT37 0QB, Northern Ireland; e-mail: {t.patterson, cd.nugent, s.zhang, i.cleland}@ulster.ac.uk

N. Khan and S. McClean are with the School of Computing and Information Engineering, Ulster University, Cromore Road, Coleraine, Co. Londonderry, BT52 1SA, Northern Ireland; e-mail: Khan-N5@email.ulster.ac.uk, si.mcclean@ulster.ac.uk

Q Ni is with the Departamento de Ingeniera Telemtica y Electrnica, Universidad Politcnica de Madrid, Carretera de Valencia km. 7, Madrid 28031, Spain; email: qin.ni@alumnos.upm.es

Manuscript received Month DD, YYYY ; revised Month DD, YYYY.

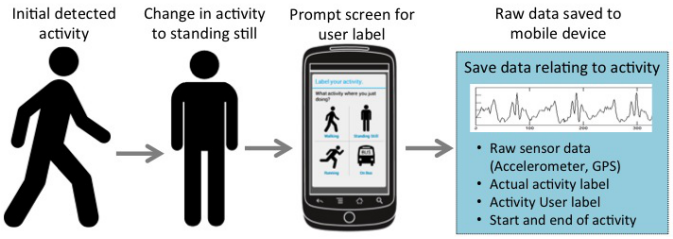


Fig. 1. Overview of the Crowd Labelling Application. The user performs an activity which is detected by the Activity Recognition module. Upon transitioning to the activity 'standing still' they are prompted for a label denoting the activity they have just transitioned from and relevant sensor data is subsequently uploaded to cloud-based storage. [6]

The autonomous, accurate detection of changes forms a fundamental component of systems that require a level of end-user engagement upon identifying transitions within an input data stream. Nevertheless, autonomously detecting changes in a data stream that correspond to a user's perception of change represents a key challenge in the overall utility and usability of such systems: prompting a user too little may diminish the impact of the application whilst prompting a user too often may negatively affect their level of engagement.

In [6] we present the Crowd Labelling Application (CLAP) which aims to facilitate the large-scale gathering of labelled activity data. This labelled data has the potential to provide the research community with an invaluable resource for training and testing activity recognition (AR) algorithms using truly representative data collected in a free-living environment. In Figure 1 an overview of the implemented framework is illustrated. This framework enables the collection and labelling of data via an Android based mobile application (app) and currently contains two primary components: an AR module and a labelling prompt module. The AR module identifies the user's current activity and contains both stationary activities, for example 'standing still' and non-stationary activities, for example 'running'. The AR module detects activities based on three second windows with a total of three consecutive windows (i.e. nine seconds of data) being required. Features within the time and frequency domains are computed for each three second window with the three consecutive windows subsequently assigned membership of an activity class using a Gaussian Mixture Model. Upon detecting a transition from an activity to 'standing still' the AR module initiates the label prompting module. This module displays a screen to the user

enabling them to click an icon representing the activity they have just transitioned from thus providing ground truth for the dataset.

Whilst the current version of CLAP enables data collection in a relatively free-living scenario there are two fundamental constraints imposed by the overall framework. Firstly, the requirement that a user transitions from an activity to 'standing still' results in potentially informative inter-activity data pertinent to real-world situations being lost. For example, the sequence {stand still - walk - jog - run - jog - walk - stand still} may be considered as a typical series of activities for running. Such inter-activity data could subsequently be utilised for training models which predict, in real-time the activity that a user is transitioning to, thus expediting the AR process by enabling the selection of appropriate classifiers. Secondly, the number of false requests for interaction that a user receives is, to some extent controlled by requiring three consecutive three second windows containing the same activity. This results in a delay between the user finishing an activity and receiving a prompt. Furthermore, this approach to controlling the number of erroneous prompts results in activities which may have an inherently short duration, for example traversing a short flight of stairs remaining undetected by the AR module.

Within this paper we extend our previous work on the Multivariate Online Change detection Algorithm (MOCA) [7] which detects changes in a multivariate data stream without requiring prior knowledge of the component distributions. MOCA provides two main benefits to user-facing applications requiring autonomous change detection such as CLAP. Firstly, MOCA includes parameters such as window size and significance values which may be adjusted depending on available computational resources and required magnitude of change. This ensures that MOCA is sufficiently generic and in an integration with CLAP would negate the requirement that a user transitions to a pre-determined activity before receiving a prompt. Secondly, MOCA can identify the commencement and cessation of an activity. This may be potentially integrated with the CLAP AR module to enable active sampling [8] of minority or previously unseen classes with the user only asked to provide labels for such classes thus maximising the utility of user interaction.

This paper offers two main contributions to knowledge: firstly we measure the utility of integrating the Benjamini Hochberg method [9] of correcting for multiple comparisons within the domain of change detection using accelerometry data; secondly, we evaluate our method from both an accuracy and real-world computational perspective against the Multivariate Exponentially Weighted Moving Average (MEWMA) and the Cumulative Sum Control Chart (CUSUM) based technique developed by Mei (2010) [10]. The accuracy evaluation was conducted using a labelled dataset obtained from 10 participants who performed 9 activities over 35 minutes each. The remainder of this paper is structured as follows: in Section II we present an overview of related work pertinent to change detection. In Section III we discuss the change detection algorithm. In Section IV we outline our data capture methodology with offline and online evaluations presented in Sections V and VI respectively. Finally, conclusions and future work are outlined in Section VII.

II. Related work

The overall aim of change point detection is to identify transitions within an input data stream that exhibit sudden changes in metrics such as mean or variance thus representing a change point in time series data [11]. Change point detection can be either classified as online or offline and is based on the target deployment and the intrinsic characteristics of an algorithm. The objective of online change detection is to monitor and process a set of data points upon arrival with a real-time constraint that processing should complete before the next set arrives. The number of new data points within successive sets depends on the target application, for example in a sliding window implementation with increments of one data point only the start and end data points will be changed. In the offline case, data is firstly collected and then analysed to detect change points. Online change detection algorithms are sequential, fast and minimise false alarms whilst, on the other hand offline change detection algorithms seek to identify all possible change points in order to attain higher levels of sensitivity (true positives) and specificity (true negatives) [12].

Within the literature there are a number of algorithms for online change point detection in sensor data. One such example is CUSUM which is considered a relatively effective approach that utilises the mean of a process for detecting small shifts. In [13] Zhang et al. (2009) a framework for detecting changes in cardiovascular events using CUSUM is presented. The core methods used in the framework are an online AR method, a biometric extraction method and a process control method used by the physiological monitoring module. In [10] a CUSUM-based change detection algorithm for multivariate data is presented that detects a change based on the global sum of locally calculated CUSUM statistics. Results are presented based on simulation that demonstrate optimality in minimising detection delays. Nevertheless, a criticism of CUSUM is that it may be inaccurate when detecting *sudden* shifts in data that are not from the same distribution [14]. The univariate change detection algorithm proposed by Jain and Wang (2014) [15] has been used to detect changes in independent random sequences. The algorithm consists of two stages: in the first phase the *most likely* change point within a processing window is identified. In the second stage the hypothesis that the most likely change point is significant is proved or disproved. The main advantage of the algorithm is that it does not require knowledge of the underlying distribution, has a small memory footprint and a relatively low computational cost.

The early drift detection method (EDDM) [16] has been developed for detecting gradual and abrupt changes in time series data. This algorithm is based on the distance between two classification errors which are computed from the average distance between two errors and the standard deviation. The change point is detected if the calculated distance is less than a threshold otherwise the new instance belongs to a previous instance. The Exponentially Weighted Moving average (EWMA) [17] has a similar concept to EDDM however, can update the estimate of error faster by using the exponentially

weighted moving average. The One-Class Support Vector Machine algorithm (OCSVM) [18] has been used for change detection in human activities. In this approach the data is modelled by a high dimensional hypersphere with changes in the radi of the hypersphere corresponding to transitions within the datastream. The Dynamic Time Warping (DTW) approach [19] is a template matching algorithm which has been used for gesture detection and biomedical signal processing. DTW utilises one template for each activity with the number of templates used directly impacting upon computational performance.

The support vector change point detection (SVCPD) algorithm utilizes the algorithm by Camci et al. (2010) [11] to detect change points in a datastream. To detect a change point, the algorithm uses the feature space, hypersphere and hypersphere radius to determine the location of new data points. The SVCPD evaluates each data point with the current hypersphere model used to identify changes within the data stream. An autoregressive model was used in Ombao et al. (2004) [20] for online change point detection in time series data and compared with likelihood based methods. The output of this technique may be poor, however, if the structure of the model differs significantly from an observed AR process. The distributional change detection algorithm [21] detects distributional changes in multivariate data. The relative entropy is used to calculate the statistical measure between two distributions and is computed using sliding windows. Information-theoretic techniques have also been used for detecting distributional changes.

Offline change detection approaches have also been used to detect changes in a time series structure. The subspace identification algorithm was proposed in [22] for change point detection in time series data. The underlying principle is that the subspace spanned by the columns of an extended observability matrix are approximately equal to those spanned by the sub-sequences of time series data. In this technique, a change point is detected based on the extended observability matrix column space and assessing if the new observation is contained in this subspace. The main advantage of this approach is that it can accommodate more abundant types of time series data as it utilises generic state-space models (SSMs) as opposed to AR models or constrained SSMs. A Bayesian algorithm was proposed by Adams and MacKay (2007) [23] for change point detection in time series data. The algorithm calculates the probability distribution of the current run with the Gaussian mean and variance used as features. The algorithm is relatively straightforward and has reasonable computational cost.

An adaptive sliding window (ADWIN) algorithm [24] has been used for change detection using a variable length window containing recently observed items. The window size is recomputed online according to the rate of change observed from the data with larger windows used for relatively static data and smaller windows used for dynamic data. The adaptive windowing algorithm is integrated with a Naive Bayes predictor to detect changes within the data stream. The Kullback-Leibler Impotence Estimation Procedure (KLIEP) [25] has been used for change point detection in time series data. KLIEP uses the density ratio estimation on population data however, this may be computationally expensive to compute in high dimensional data. The primary advantages of KLIEP are its convergence properties and automatic model selection however, there are criticisms surrounding its speed, robustness and that it has convex optimisation problems. A change detection method with feature selection for high dimensional time series data has been proposed in [26] and is termed the additive Hilbert-Schmidt Independence Criterion (aHSIC). It uses the weighted sum of HISC scores between features and their corresponding binary labels. The HSIC is also known as a kernel based independence measure because it uses feature selection during its detection measure estimation enabling the method to use features which are most indicative of an abrupt change.

Overall, many algorithms within the literature require prior knowledge about the characteristics of change points and the data stream's underlying distributions which may result in them being ineffective for our target application. Additionally, most require the observation of numerous estimation parameters and also necessitate the tuning of parameters prior to execution with a primary emphasis on accuracy rather than efficiency. Given our exemplar target application of soliciting data labels from users via a smartphone app upon commencement of a new activity we require an algorithm that has a low computational and memory footprint, does not require prior knowledge regarding a data stream's underlying distributions and can be executed in real-time. These requirements ensure that the algorithm can be executed on limited computational hardware such as a smartphone, can operate in diverse scenarios for multiple users with potentially different activity characteristics and can utilize the most recent sensor observations.

III. Multivariate Online Change detection Algorithm (MOCA)

Within this section we present the Multivariate Online Change detection Algorithm (MOCA) which does not require prior knowledge of the underlying data distribution(s) and can be executed in real-time. MOCA autonomously detects changes in an input data stream such as accelerometry data as follows: consider a data stream of length q consisting of data points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q$. Each data point $\mathbf{x}$ is a B element vector where B is the number of sensor observations for each variable. The data stream may contain points from multiple distributions, for example $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{k-1}$ may have distribution D_1 whilst $\mathbf{x}_k, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_q$ may have distribution D_2. It is therefore the overall aim of the algorithm to identify the position in the data stream of change points k ($1 \leq k \leq q$).

MOCA follows an hypothesis-and-verification principle: in the hypothesis step a point is detected within the window under consideration which maximises the test statistic. In the second stage the hypothesis that a detected change point is significant is verified.

A. Hypothesis Generation

In the hypothesis generation stage we pass an analysis window of length n over the data stream assuming that there is

a maximum of one change point per window. The movement of the window over the data stream may be either distinct in which case the start of a new window (other than the first) is at position $m + cn + 1$ where m is the padding size and c is the number of previous windows. Alternatively, a sliding window version of the algorithm may be executed with the start position incremented by a predetermined number of data points. For ease of notation we denote the data points within a window as $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ regardless of their actual position within the data stream. Following Jain and Wang [15] we pad either side of the window with m data points such that the analysis window contains data points $\mathbf{x}_{1-m}, \ldots, \mathbf{x}_{n+m}$ therefore containing a total of $n+2m$ data points. This padding ensures sufficient data to compute summary statistics at the window extremities and is particularly crucial when executing a distinct window version of the algorithm; however, results in a minimum latency of m data points.

Within each window we slide an index variable, l, $1 < l \leq n$ subsequently computing summary statistics of the component distributions separated at l. Specifically, we compute the means, $\bar{\mathbf{f}}_1(l)$ and $\bar{\mathbf{f}}_2(l)$, which contain the mean of observations, in addition to variance-covariance matrices, $\mathbf{S}_1(l)$ and $\mathbf{S}_2(l)$, which contain the variance of observations in the diagonals and their covariance in the off-diagonals. To ensure that the change detection algorithm can operate in online scenarios we compute $\bar{\mathbf{f}}_1(l)$, $\bar{\mathbf{f}}_2(l)$ and $\mathbf{S}_1(l)$, $\mathbf{S}_2(l)$ recursively. Thus as index l increments to position $l + 1$ the summary statistics are calculated as follows (equations 1-4):

$$\bar{\mathbf{f}}_1(l+1) = \frac{m+l-1}{m+l}\bar{\mathbf{f}}_1(l) + \frac{f(\mathbf{x}_{l+1})}{m+l}, \tag{1}$$

$$\bar{\mathbf{f}}_2(l+1) = \frac{n+m-l+1}{n+m-l}\bar{\mathbf{f}}_2(l) - \frac{f(\mathbf{x}_{l+1})}{n+m-l}, \tag{2}$$

$$\mathbf{S}_1(l+1) = \frac{m+l-1}{m+l}\mathbf{S}_1(l) + \frac{1}{m+l-1} \times [\mathbf{x}_{l+1} - \bar{\mathbf{f}}_1(l+1)]^{'}[\mathbf{x}_{l+1} - \bar{\mathbf{f}}_1(l+1)], \tag{3}$$

$$\mathbf{S}_2(l+1) = \frac{n+m-l+1}{n+m-l}\mathbf{S}_2(l) - \frac{1}{n+m-l} \times [\mathbf{x}_{l+1} - \bar{\mathbf{f}}_2(l+1)]^{'}[\mathbf{x}_{l+1} - \bar{\mathbf{f}}_2(l+1)], \tag{4}$$

where $f(\mathbf{x}_{l+1})$ is the value of the datastream at position $l+1$. Having calculated summary statistics before and after l we proceed to compute the F statistic at position l, F_l as follows [27]:

$$F_l = \frac{n_1 + n_2 - B - 1}{B(n_1 + n_2 - 2)}T^2, \tag{5}$$

where $n_1 = m+l-1$, $n_2 = n+m-l+1$, B is the number of variables and T^2 is the Hotelling T-squared statistic calculated as [27],

$$T^2 = (\bar{\mathbf{f}}_1 - \bar{\mathbf{f}}_2)^{'}\left\{\mathbf{S}_p\left(\frac{1}{n_1} + \frac{1}{n_2}\right)\right\}^{-1}(\bar{\mathbf{f}}_1 - \bar{\mathbf{f}}_2), \tag{6}$$

where $\mathbf{S}_p$ is the pooled variance-covariance matrix,

$$\mathbf{S}_p = \frac{(n_1 - 1)\mathbf{S}_1 + (n_2 - 1)\mathbf{S}_2}{n_1 + n_2 - 2}. \tag{7}$$

Under the null hypothesis (i.e. equal distributions) and assuming Gaussian distributions this has an F distribution [27]. We choose the point l which maximises F_l as the most likely change point within a window and proceed to the hypothesis verification phase.

B. Hypothesis Verification

A hypothesis verification stage is executed to prove or disprove the null hypothesis that a significant change did not occur at point l. Firstly, we compute the probability of finding an F value lower than that calculated in Equation 5 resulting in d. The F Cumulative Distribution Function is utilised for this phase with B and $n_1 + n_2 - b$ degrees of freedom. The test's p-value is then computed as,

$$p = 1 - d. \tag{8}$$

In the sliding window version of the algorithm the hypothesis verification stage is conducted multiple times for a set of datapoints which will have changed only by the window's increment value, i.e. only the start and end data points will have been replaced. Thus in our previous we work we used Bonferroni correction [28] to compute an updated threshold t,

$$t = \alpha/n. \tag{9}$$

This adjusts the original confidence value, α to reflect the confidence for the entire window of size n and not a single, isolated value with the null hypothesis rejected if $p < t$. The Bonferroni method controls the familywise error rate (FWER) where, in our approach a 'family' is a group of statistical tests of size equal to the window size, when using the sliding window version of the algorithm with increments of one datapoint. This method seeks to reduce the probability of even one false detection. As reported in our previous work [7] this results in datapoints within larger windows requiring a higher test statistic in order to reject the null hypothesis subsequently leading to an increase in false negatives. This is in agreement with studies in alternative domains such as Conservation Genetics which also found Bonferroni correction to provide an overly conservative adjusted confidence value when using larger sample sizes [29].

An alternative approach to correct for multiple comparisons is to control the false discovery rate (FDR). FDR procedures control the *proportion* of Type I errors (false positives) within a group of tests as opposed to controlling the FWER which is the probability of *at least one* false positive. Thus, methods controlling the FDR have greater sensitivity at the potential cost of increased Type I errors. A popular technique within the literature for controlling the FDR is proposed by Benjamini and Hochberg (1995) [9]. This method compares p-values against their Benjamini-Hochberg critical value. It subsequently results in an error rate that is equivalent to the FWER when all the null hypothesis are true but is smaller otherwise [9]. The Benjamini-Hochberg method is implemented as described in Algorithm 1. In the sliding window version of the algorithm with increments of one datapoint the set of tests, T will be of equal size to the window size, n.

Algorithm 1 Benjamini-Hochberg method for controlling False Discovery Rate

Require: A set of tests, T

1: Sort associated p-values, T_p into ascending order such that $T_{pi} < T_{pi+1}$ where i is an index variable, $i = 1 \dots n$ and n is the sample size.
2: **for** $i = 1 : n$ **do**
3: Compute Benjamini-Hochberg critical value associated with each p-value,

$$T_{bi} = (i/n)Q, \tag{10}$$

where i is the index variable, n is the number of samples and Q is the chosen false discovery rate, $0 < Q < 1$.
4: **end for**
5: Find the index of the largest p-value, T_{maxi} that is less than the associated Benjamini-Hochberg critical value, i.e. $T_{pi} < T_{bi}$.
6: Label all p-values as significant that have sorted index $i < T_{maxi}$.

IV. Experimental setup

To enable the quantitative evaluation of human activity change detection algorithms a dataset was collected using 10 healthy participants (three female, seven male). Each participant wore a Shimmer sensing platform [30] on their wrist to represent a smart watch with a sampling frequency of 102.4 Hz. Additional Shimmers were located on the participant's sternum and lower limb to facilitate future work analysing the optimal sensor placement for detecting changes in human activity.

Each participant performed a total of nine distinct sedentary and non-sedentary activities as described in Table I. The activities were categorised as either static indicating that the participant was asked to remain comfortably still, i.e. small natural movements were permitted; transitional indicating that the data captures the transition between two activities; and, dynamic indicating that the activity inherently contains purposeful human movement.

Change points were manually labelled by a human expert based on the recorded time a participant was asked to change activity in addition to a visual inspection of the sensor data. In the event of a participant being unable to complete the tasks in succession they were permitted to rest with the start and end time recorded and the relevant sensor data subsequently removed from the dataset. The resultant dataset contains a continuous data stream of approximately 35 minutes for each participant with the activities performed according to the order in Table I. There are 95 labelled transitions for each participant, i.e. 950 in total. The majority of transitions are from static to dynamic or vice versa, however the dataset also contains transitions from dynamic to dynamic, for example walking to running.

V. Offline evaluation

Within this Section we present results (accuracy, precision, sensitivity, specificity) based on executing MOCA, MEWMA and a CUSUM-based algorithm[1] developed by Mei [10] on the dataset captured following the protocol described in Section IV; this ensures that each algorithm is evaluated using exactly the same data. The results are based on the true positive (TP), true negative (TN), false positive (FP), false negative (FN) rates and were obtained using a Matlab implementation of the algorithms. We evaluate the impact provided by controlling the FDR (MOCA (Benjamini Hochberg)) as opposed to the FWER (MOCA (Bonferroni)) and additionally compare our algorithm against MEWMA and Mei as they provide a similarly light-weight approach to detecting changes in an input data stream.

A. Multivariate Exponentially Weighted Moving Average

The EWMA is a statistical method for observing a process that averages the input data based on a points position within a datastream such that less weight is given to older data points [31]. The primary aim is to quickly detect small shifts based on the EWMA statistic which is an exponentially weighted moving average of all prior data. The MEWMA is an extension of EWMA for multivariate data [32] and is defined as:

$$\mathbf{Z}_i = \mathbf{R}\mathbf{X}_i + (1 - \mathbf{R})\mathbf{Z}_{i-1}, \tag{11}$$

where $\mathbf{Z}_i$ is the i^{th} MEWMA vector, $\mathbf{R}$ is a diagonal matrix with weighting parameters $\lambda_1, \lambda_2, \dots, \lambda_p$ on the main diagonals where p is the number of variables and $0 < \lambda \leq 1$ and $\mathbf{X}_i$ is the i^{th} observation vector, $i = 1, 2 \dots n$. The MEWMA chart identifies an out-of-control signal when,

$$T_i^2 = (\mathbf{X}_i - \mu_0)' \Sigma_{\mathbf{Z}_i}^{-1} (\mathbf{X}_i - \mu_0) > h, \tag{12}$$

where $\Sigma_{\mathbf{Z}_i}$ is the variance-covariance matrix of $\mathbf{Z}_i$, μ_0 is the on-target mean vector and $h(> 0)$ is chosen to achieve a specified in-control.

B. Sum of CUSUM

In [10] a scalable scheme for online change detection is proposed based on the sum of local cumulative sums calculated in multiple datastreams. Results are presented based on numerical simulations that demonstrate asymptotic optimality in minimizing detection delays for all combinations of affected data streams subject to a global false alarm constraint. The change detection algorithm by Mei [10] identifies a change as follows: at time Δ, for each datastream the local CUSUM statistic $W_{\Delta}^{(b)}$ is calculated recursively, where b is the datastream under consideration and B is the total number of datastreams. The stopping time (i.e. the time at which a datapoint is labelled as a change), $N_{sum}(t)$, is defined as,

$$N_{sum}(t) = inf \left\{ \Delta \geq 1 \sum_{b=1}^{B} W_{\Delta}^{(b)} \geq t \right\}, \tag{13}$$

where t is a predefined threshold.

The dataset collected according to the protocol described in Section IV is multivariate in nature thus requiring change

[1] Subsequently referred to as 'Mei'.

TABLE I
OVERVIEW OF ACTIVITIES IN DATASET

Execution Order	Label	Type	Description
1	Standing	Static	Stand for 5 minutes (min)
2	Stand-sit	Transitional	Stand for 10 seconds (s), sit for 10s (15 repetitions)
3	Sleeping	Static	Lie on sofa for 5 min
4	Stand-walk	Transitional	Stand for 10s, walk for 20s (15 repetitions)
5	Sit-lie	Transitional	Sit for 10s, lie for 10s (15 repetitions)
6	Walking	Dynamic	Walk on treadmill at constant speed for 5 min
7	Running	Dynamic	Run on treadmill at constant speed for 5 min
8	Watching TV	Static	Sit on sofa for 5 min
9	Vacuum	Dynamic	Vacuum for 5 min

detection algorithms to analyse observations with more than one variable, for example x, y, z accelerometry values. Additionally to ensure timely change detection for applications such as CLAP that require user engagement it is necessary that developed algorithms can operate in real-time. In comparison to many of the change detection algorithms and concepts described in related work (Section II) MEWMA has been utilised in real-time applications to autonomously detect changes in human physiological data such as electroencephalography (EEG) [33] and transitions between locations based on smartphone movements [34]. Similarly, the algorithm developed by Mei has been utilised for autonomously detecting shutdown periods in chemical plants by monitoring sensor values such as temperature and pressure in real-time. [35]. Thus, the properties of being multivariate in nature, not requiring prior knowledge of the input datastream's distribution and being lightweight i.e. can be implemented to execute in real-time on mobile devices results in MEWMA and Mei being directly comparable with our approach (MOCA) whilst being representative of alternative change detection approaches i.e. weighted moving averages and cumulative sum.

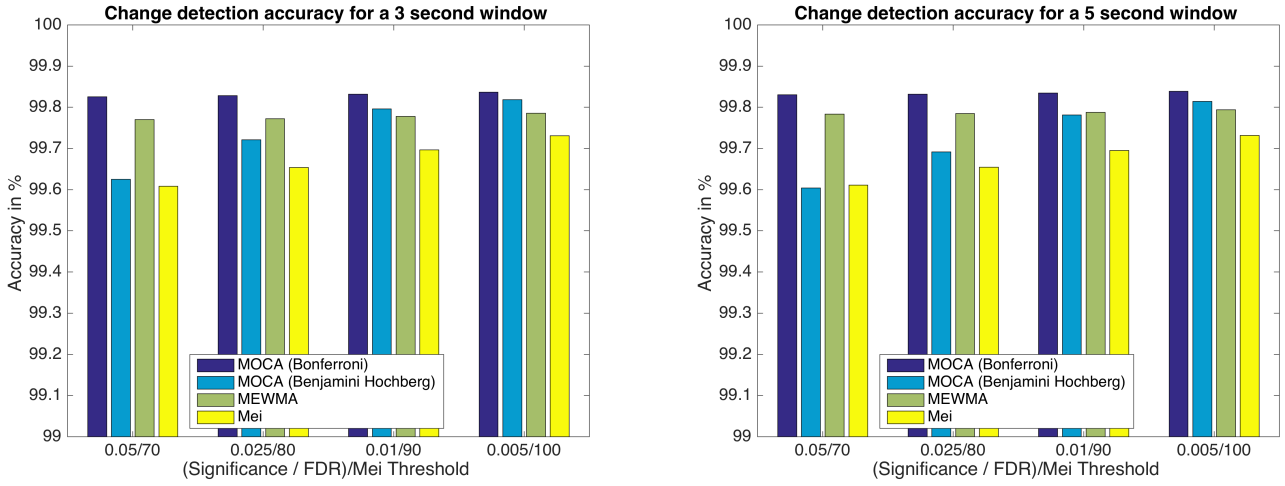
C. Evaluation

We define the positive and negative detection cases as follows: a TP is a correctly identified change. When determining true positives a one second margin of error was included at either side of the manually labelled change point. This was based on half a second to accommodate any subjectivity present in the manual labelling in addition to half a second to accommodate variation in the human reaction time between receiving an auditory command and commencing a motor task [36] [37]. Thus, a detected change point was considered true if its index in the data stream, $l \in \{z-(f) \dots z+(f)\}$ where z is the index in the data stream of the manually labelled change point and f is the sampling frequency in Hz. Furthermore, when the sensor data at a change point is examined it can be seen that the data values increase or decrease over a range of data points. In the sliding window version of the algorithm this results in multiple detected changes that are in close proximity or sequential albeit indicitive of the same 'event'. We therefore incorporate a refractory period [38] that represents the minimum interval between two changes in human activity. Within this work we have utilised a fixed refractory period of one second. We define a TN as a non-transitional point which is not labelled as a change.

A FP is a non-transitional point which is highlighted by the algorithm as a change in activity. In terms of user experience this type of error is likely to be the most detrimental as it will result in them receiving unintuitive requests for interaction. A FN occurs when the algorithm fails to detect a change in the user's activity. Bearing in mind our target application this type of error would primarily impact upon the quality of the dataset labels as the labelling program would not request user interaction.

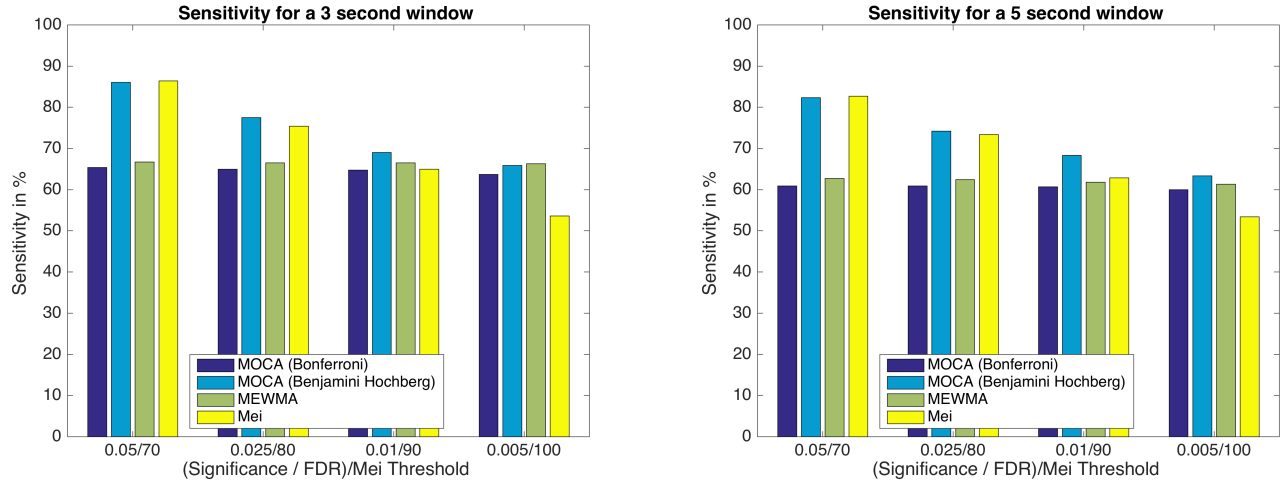
As the target application relies on identifying changes in a user's activity we experiment with window sizes based on those suggested within AR literature specifically three and five seconds [39]. Within [39] and [40] the authors propose windows with no overlap and 50% overlap respectively for feature extraction with window sizes of several seconds used to ensure that a sufficient number of activity cycles are contained within a window, for example walking or running. For the results presented within this Section we use a sliding window with increments of one data point to ensure that a labelled change is detected as soon as possible. Autonomously determining an appropriate window increment will form part of future work and is likely to be dependent on the computational resources available in addition to the user's profile for example, their typical daily routine such as sleeping patterns and device state for example, battery level. For MOCA and MEWMA we experiment with four significance/FDR values: 0.05, 0.025, 0.01 and 0.005 whilst for Mei we utilize empirically observed thresholds of 70, 80, 90 and 100.

In Figures 2 to 5 results are shown based on executing MOCA with Bonferroni correction, MOCA with Benjamini Hochberg correction, MEWMA and Mei on the labelled accelerometry data. Overall accuracy defined as $TP + TN/(TP + TN + FP + FN)$ (Figure 2) was relatively high given the disproportionate number of TNs in the data. Accuracy ranged from 99.84% for MOCA (Bonferroni) to 99.60% for MOCA (Benjamini Hochberg). From an accuracy perspective MOCA outperformed MEWMA for all significance / FDR values and similarly outperformed Mei for 7 of the 8 experiments. Utilizing Benjamini Hochberg correction resulted in lower overall accuracy than Bonferroni correction. This was caused by the Benjamini Hochberg procedure providing a less conservative correction than Bonferroni which ultimately resulted in a higher number of TPs at the expense of increased FPs. Reducing the significance / FDR and increasing the Mei threshold led to an increase in accuracy for all algorithms. This was caused by hypothesised change points requiring a



(a) (b)

Fig. 2. Comparison of overall accuracy between MOCA (Bonferroni), MOCA (Benjamini Hochberg), MEWMA and Mei using 3 (a) and 5 (b) second windows.



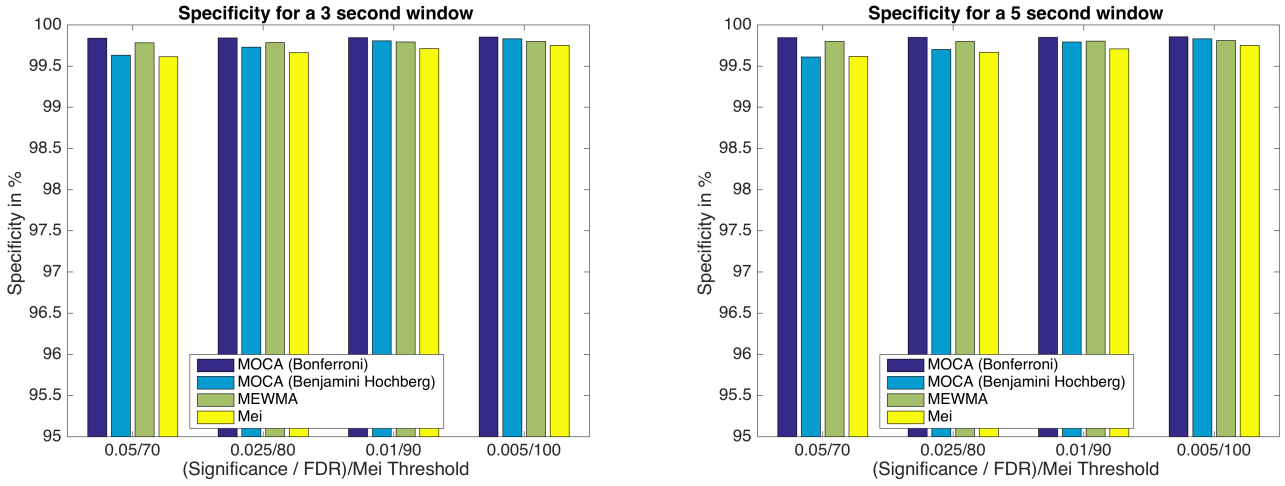
(a) (b)

Fig. 3. Comparison of sensitivity between MOCA (Bonferroni), MOCA (Benjamini Hochberg), MEWMA and Mei using 3 (a) and 5 (b) second windows.

higher test statistic in order to reject the null hypothesis which resulted in less FPs.

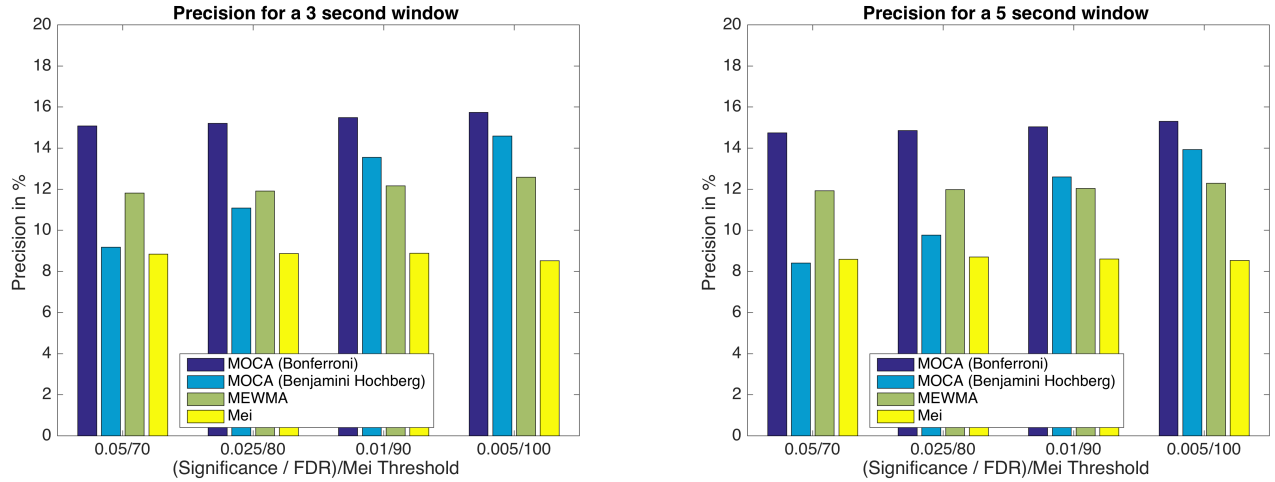
The sensitivity defined as $TP/(TP+FN)$ is illustrated in Figure 3. The maximum sensitivity was 86.38% (Mei) and the minimum was 53.43% (Mei). This is intuitive as the implementation of MOCA and MEWMA utilises all points within the window under consideration, i.e. the data points either side of the sliding index variable are used when determining if a hypothesised change is significant. On the other hand, Mei only considers data points from the left most extremity of the window up to the index variable thus making it more sensitive to small variations across all sensor readings. The sensitivity of each algorithm decreased as the significance / FDR was decreased and the Mei threshold increased due to a higher number of FNs. From the standpoint of collecting user labelled activity data the sensitivity of the algorithm signifies the magnitude of change in accelerometry required before interaction would be requested. In Figure 4 the specificity of each algorithm defined as $TN/(TN+FP)$ is illustrated. The minimum specificity measured was 99.61% (MOCA (Benjamini Hochberg)) with a maximum of 99.86% (MOCA (Bonferroni)). As the significance / FDR was decreased and the Mei threshold increased the specificity of each algorithm increased caused by a reduced number of FPs.

The overall precision (Figure 5) defined as $TP/(TP+FP)$ was $< 20\%$ for all algorithms with MOCA (Bonferroni) achieving higher precision than MEWMA and Mei for each window size and significance / Mei threshold values. The relatively low precision results were caused by the number of false positives in the results, for example when utilizing MOCA (Bonferroni) on a window size of five seconds with a significance of 0.05 there were a total of 575 TPs and 3,328 FPs. From a user's perspective the precision represents the proportion of correct requests for interaction. Whilst our



(a) (b)

Fig. 4. Comparison of specificity between MOCA (Bonferroni), MOCA (Benjamini Hochberg), MEWMA and Mei using 3 (a) and 5 (b) second windows.



(a) (b)

Fig. 5. Comparison of precision between MOCA (Bonferroni), MOCA (Benjamini Hochberg), MEWMA and Mei using 3 (a) and 5 (b) second windows.

approach (MOCA) outperforms MEWMA and Mei for both window sizes and significance / threshold values a key part of future work will be to increase the precision of MOCA. One approach may be to integrate a two-step approach whereby user interaction is only requested if a hypothesised change is determined significant and an additional criteria is met for example, the number of times a data point is identified as a significant change by successive sliding windows.

Overall MOCA (Bonferroni) outperformed MEWMA and Mei for accuracy, specificity and precision metrics whilst MOCA (Benjamini Hochberg) provided higher sensitivity levels than MOCA (Bonferroni) and MEWMA. MOCA (Benjamini Hochberg) provided higher sensitivity levels than Mei for FDR and threshold values less than 0.05 and greater than 70 respectively. As the window size increased from three to five seconds the accuracy increased for all algorithms as there was a reduced number of FPs. This was caused by the formulation of Equations 9 (Bonferroni) and 10 (Benjamini-Hochberg) which compute a threshold based on the window size thus requiring change points within larger windows to have a higher test statistic in order to reject the null hypothesis.

It is envisaged that the selection of MOCA algorithms will be dependent upon the objective of the application. Where a trade-off is required between accuracy, sensitivity and specificity with an emphasis on a higher TP rate MOCA (Benjamini Hochberg) correction for a five second window with an FDR of 0.005 provides a reasonable solution at the expense of an increase in FPs. On the other hand if the application emphasis is on a higher TN rate MOCA (Bonferroni) correction for a five second window and a significance threshold of 0.005 provides a reasonable solution at the expense of an increase in FNs. Given our target application of soliciting user interaction to obtain a labelled dataset these results indicate that MOCA (Bonferroni) is the most suitable algorithm in order to minimise spurious requests for interaction; however, a key part of future work will focus on improving the precision of

MOCA.

D. Latency

The latency of the change detection algorithm refers to the elapsed time between a transition in a user's activity occurring and a detected change. Latency is calculated as the number of datapoints from a TP to the end of an analysis window of size $n + 2m$ datapoints. It is useful to note that in a free living scenario the datapoint at the right hand extremity of the analysis window represents the most recent measurement with the number of new datapoints consistent with the increment value used, i.e. a sliding or distinct window. The padding region, m is utilised to ensure that there is sufficient data to perform calculations at window extremities; however, results in a minimum latency, equal to size of m. In this work a static size was used for each buffer region with the length of m set to one second. Determining an optimum value for m is likely to be dependent upon sampling frequency and target application and will form part of future work.

In Tables II and III mean, μ and standard deviation, σ latency values are presented for three and five second windows. Across the four algorithms evaluated an increase in window size from three to five seconds led to an increase in the μ and σ latency values. In Equation 7 the number of points *before*, n_1 and *after*, n_2 an hypothesised change point, l is used to scale the variance-covariance matrices to calculate a pooled variance-covariance matrix. Thus, the number of points *after* l need to be sufficient to ensure that the values of the variance-covariance matrix are represented in the pooled variance-covariance matrix. Furthermore, the points *after* l must represent a distinct change from those *before*.

A consistent trend in latency was not observed as the significance/FDR decreased and the Mei threshold increased. This may be caused by real transitions in human activity being significant for very high confidence values. Across the four algorithms Mei had the lowest overall latency for all window sizes and significance values. When a transition in human activity occurs the sensor values increase or decrease over a range of data points with each successive data point becoming increasingly indicative of a change. As previously mentioned the implementation of MOCA and MEWMA considers all data points in a window whilst the algorithm by Mei only considers data points up to the sliding index variable. Thus, Mei is sensitive to small changes in values that occur across all sensors and subsequently TPs are detected earlier than the alternative algorithms considered.

VI. Online Evaluation

Within this Section we present empirical timings for a smartphone based implementation of MOCA thus demonstrating its potential to operate in online scenarios on limited computational hardware. The change detection algorithm presented in Section III was implemented in Java and targeted at Android versions 4.1 and above. The Android platform was chosen as it currently has the largest global market share [41] resulting in a higher number of potential users.

TABLE IV
Device Setup for Capturing Execution Timings

Attribute	Value
Device	Motorola Moto G (2nd Generation)
Android Version	5 (Lollipop)
WIFI	On - Connected to test access point
Bluetooth	On - Not connected to any devices
Location	On
Cellular Network	Off - No SIM in phone
Screen	Off - Device locked during tests
Applications Open	Gmail, Chrome
Sensors Polled	Accelerometer
Sensor Delay	20 milliseconds

A second generation Motorola Moto G smartphone [42] was utilised as the hardware specification can be considered representative of a low to mid-range smartphone. The device was set up as specified in Table IV with parameter choices designed to be representative of a typical user. Primary smartphone functionality such as WiFi, Bluetooth and location services were activated. The cellular network was unavailable as the device did not have a SIM card installed. The screen was turned off during tests by locking the device which would typically occur when a user carries the smartphone in their pocket. The Gmail mailbox client and Chrome web browser opened at a static webpage (www.bbc.co.uk/news) and remained running in a background state.

To ensure consistency with the data captured following the protocol described in Section IV we only polled the accelerometer. A sensor delay rate of 20 milliseconds was used which signifies the maximum interval between sensor events dispatched to the change detection application. This results in a sampling frequency of approximately 50 Hz.

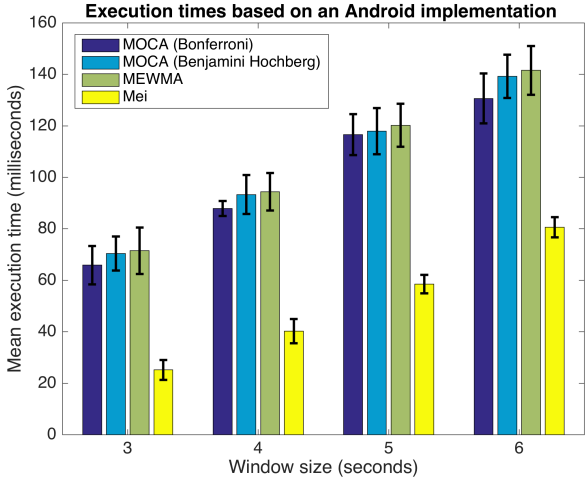


Fig. 6. Comparison of empirical timings between MOCA (Bonferroni), MOCA (Benjamini Hochberg), MEWMA and Mei

To capture execution timings the device was set up as specified in Table IV. The device was placed on a desk for all experiments and each change detection algorithm executed for five minutes. In Figure 6 empirical mean execution times are shown for window sizes of three, four, five and six seconds with the standard deviation represented as an error

TABLE II
MEASURED LATENCY IN SECONDS FOR A 3 SECOND WINDOW OBTAINED BY EXECUTING MOCA (BONFERRONI), MOCA (BANJAMINI HOCHBERG), MEWMA AND MEI. NOTE: A ONE SECOND BUFFER WAS INCLUDED AT WINDOW EXTREMITIES.

	MOCA (Bon.)	**MOCA (Ben.)**	**MEWMA**	**Mei**
(Sig./FDR)/Thresh.				
0.05/70	$\mu = 2.29\ \sigma = 0.43$	$\mu = 2.28\ \sigma = 0.52$	$\mu = 2.34\ \sigma = 0.42$	$\mu = 1.01\ \sigma = 0.49$
0.025/80	$\mu = 2.30\ \sigma = 0.43$	$\mu = 2.29\ \sigma = 0.50$	$\mu = 2.35\ \sigma = 0.42$	$\mu = 1.05\ \sigma = 0.55$
0.01/90	$\mu = 2.30\ \sigma = 0.42$	$\mu = 2.27\ \sigma = 0.43$	$\mu = 2.37\ \sigma = 0.42$	$\mu = 1.12\ \sigma = 0.62$
0.005/100	$\mu = 2.36\ \sigma = 0.42$	$\mu = 2.31\ \sigma = 0.44$	$\mu = 2.40\ \sigma = 0.41$	$\mu = 1.24\ \sigma = 0.58$

TABLE III
MEASURED LATENCY IN SECONDS FOR A 5 SECOND WINDOW OBTAINED BY EXECUTING MOCA (BONFERRONI), MOCA (BANJAMINI HOCHBERG), MEWMA AND MEI. NOTE: A ONE SECOND BUFFER WAS INCLUDED AT WINDOW EXTREMITIES.

	MOCA (Bon.)	**MOCA (Ben.)**	**MEWMA**	**Mei**
(Sig./FDR)/Thresh.				
0.05/70	$\mu = 2.58\ \sigma = 0.80$	$\mu = 2.39\ \sigma = 0.76$	$\mu = 2.60\ \sigma = 0.79$	$\mu = 1.61\ \sigma = 1.26$
0.025/80	$\mu = 2.59\ \sigma = 0.82$	$\mu = 2.47\ \sigma = 0.79$	$\mu = 2.58\ \sigma = 0.77$	$\mu = 1.64\ \sigma = 1.30$
0.01/90	$\mu = 2.57\ \sigma = 0.80$	$\mu = 2.43\ \sigma = 0.67$	$\mu = 2.58\ \sigma = 0.77$	$\mu = 1.78\ \sigma = 1.38$
0.005/100	$\mu = 2.64\ \sigma = 0.86$	$\mu = 2.47\ \sigma = 0.66$	$\mu = 2.62\ \sigma = 0.78$	$\mu = 1.54\ \sigma = 1.29$

bar. Intuitively the execution time increases with window size with observed mean times ranging from approximately 21 milliseconds (Mei) for a three second window to approximately 135 milliseconds for a six second window (MEWMA). Overall Mei had the fastest execution time as it primarily involves summation (Equation 13). MOCA (Benjamini Hochberg) and MEWMA have a slower mean execution time than MOCA (Bonferroni) due to increased computation when executing Algorithm 1 and Equation 11 respectively. The main source of variability is likely to be caused by background system processes such as 'system_server' which provides context for other system services and 'sensors.qcom' which manages the device's sensors.

When executing MOCA on a smartphone in a real-time free living scenario there is likely to be two main effects caused by constraints imposed by the hardware. Firstly, the latency between a change occurring and its subsequent detection will be influenced both by the potential sensor sampling rates in addition to the device's processing power and its current load. One approach to mitigating this effect may be to implement dynamic window increments that are chosen based on the current mean algorithm execution time. Secondly, the offline tests discussed in Section V were performed using a computer with a 64-bit processor. It is therefore possible that the difference in representing values between 64-bit and 32-bit processing architectures may be sufficient to have an effect on the accuracy, precision, sensitivity and specificity of MOCA; however, this effect would also be observed for many alternative algorithms such as Mei. Overall, the timings indicate that MOCA can operate in real-time on limited computational hardware such as a smartphone.

VII. CONCLUSIONS AND FUTURE WORK

In this paper we have presented an algorithm for detecting changes in human activity based on accelerometry data. The algorithm utilizes the mean and variance-covariance of sensor values and, in comparison to placing a static threshold on these values enables the required magnitude of a change to be set via significance values. This adaptive approach ensures that MOCA is sufficiently generic for a range of change detection applications without requiring prior knowledge about the characteristics of the underlying data streams. Detected changes can be subsequently utilised by applications such as CLAP to solicit user interaction or within an activity classification framework by influencing the starting position of processing windows. In this paper we have extended our previous work on MOCA by exploring the utility of controlling the FDR as opposed to the FWER and additionally by evaluating the algorithm against MEWMA and Mei from both an accuracy and computational perspective. Overall, MOCA outperformed MEWMA and Mei achieving higher accuracy, sensitivity, specificity and precision. Controlling the FDR as opposed to FWER resulted in higher sensitivity at the expense of accuracy, precision and specificity. This was intuitive as controlling the FDR provides a less conservative correction than the FWER. The algorithm by Mei had the lowest measured latency whilst MOCA (Benjamini Hochberg) achieved lower latency than both MOCA (Bonferroni) and MEWMA. Execution results were based on an Android implementation of MOCA, MEWMA and Mei and demonstrated that the execution time increased with window size.

There will be two main areas of future work: firstly we will conduct a small scale ($N \approx 10$) data collection via an Android app with users asked to manually label when they have changed activity. Following our previous work [6] we will follow a semi-structured experimental protocol whereby participants will be asked to perform a high-level task such as 'walk to shop' and are shadowed by a trained observer who also records when the participant changes activity. The primary motivation behind this phase is to gather a labelled dataset from all available sensors on the Motorola Moto G smartphone (Section VI). This will facilitate the empirical selection and further evaluation of MOCA parameters such as window size, window increments and significance / FDR values on data containing labels focused towards a user's perception of change. Secondly, a key part of future work will focus on enhancing the precision of MOCA using for example, a voting mechanism whereby the number of times a

data point is identified as a significant change by successive windows is incorporated when rejecting the null hypothesis. Overall, MOCA represents a viable approach to autonomously detecting transitions such as changes in human activity from sensor data and with the incorporation of identified future work has the potential to become an integral component of smartphone applications such as CLAP.

ACKNOWLEDGMENT

The authors gratefully acknowledge support from Invest Northern Ireland under research and development grant RD0513844 and the Engineering and Physical Sciences Research Council through the Multidisciplinary Assessment of Technology Centre for Healthcare (EP/F063822/1 and EP/G012393/1). Invest Northern Ireland also partially supported this project under the Competence Centre Program Grant RD0513853 - Connected Health Innovation Centre.

REFERENCES

[1] M. Basseville and I. Nikiforov, *Detection of Abrupt Changes: Theory and Application*, informatio ed. Englewood Cliffs, NJ: Prentice Hall, 1993.

[2] P. Sarkar and W. Q. Meeker, "A bayesian on-line change detection algorithm with process monitoring applications," *Quality Engineering*, vol. 10, no. 3, pp. 539–549, 1998.

[3] E.-V. Pikoulis and E. Psarakis, "Automatic seismic signal detection via record segmentation," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 53, no. 7, pp. 3870–2884, July 2015.

[4] D. Clifton, D. Wong, L. Clifton, S. Wilson, R. Way, R. Pullinger, and L. Tarassenko, "A Large-Scale Clinical Validation of an Integrated Monitoring System in the Emergency Department," *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 4, pp. 835–842, Jul. 2013.

[5] Q. Ni, T. Patterson, I. Cleland, and C. Nugent, "Dynamic detection of window starting positions and its implementation within an activity recognition framework." *Journal of biomedical informatics*, vol. 62, pp. 171–180, Jul. 2016. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/27392647

[6] I. Cleland, M. Han, C. Nugent, H. Lee, S. McClean, S. Zhang, and S. Lee, "Evaluation of prompted annotation of activity data recorded from a smart phone." *Sensors*, vol. 14, no. 9, pp. 15 861–79, Jan. 2014.

[7] T. Patterson, S. McClean, C. Nugent, S. Zhang, L. Galway, and I. Cleland, "Online Change Detection for Timely Solicitation of User Interaction," in *Ubiquitous Computing and Ambient Intelligence. Personalisation and User Adapted Services*, R. Hervás, S. Lee, C. Nugent, and J. Bravo, Eds. Belfast, Northern Ireland: Springer, 2014, pp. 116–123.

[8] B. Settles, *Active Learning: Synthesis Lectures on Artificial Intelligence and Machine Learning*, 1st ed., R. J. Brachman, W. W. Cohen, and T. Dietterich, Eds. San Rafael, California: Morgan & Claypool, 2012.

[9] Y. Benjamini and Y. Hochberg, "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 57, no. 1, pp. 289–300, 1995.

[10] Y. Mei, "Efficient scalable schemes for monitoring a large number of data streams," *Biometrika*, vol. 97, no. 2, pp. 419–433, Apr. 2010. [Online]. Available: http://biomet.oxfordjournals.org/cgi/doi/10.1093/biomet/asq010

[11] F. Camci, "Change point detection in time series data using support vectors," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 24, no. 01, pp. 73–95, 2010.

[12] A. B. Downey, "A novel changepoint detection algorithm," *arXiv preprint arXiv:0812.1237*, 2008.

[13] S. Zhang, L. Galway, S. McClean, B. Scotney, D. Finlay, and C. Nugent, "Deriving Relationships between Physiological Change and Activities of Daily Living using Wearable Sensors," in *Sensor Systems and Software*. Miami, Florida, USA: Springer Berlin Heidelberg, 2011, pp. 235–250.

[14] D. Prajapati and P. Mahapatra, "A new X chart comparable to CUSUM and EWMA charts," *International Journal of Productivity and Quality Management*, vol. 4, no. 1, pp. 103–128, 2009.

[15] A. Jain and Y.-f. Wang, "A New Framework for On-Line Change Detection (Unpublished), Accessed October 2016," Online, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.62.5929.

[16] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method," in *Fourth International Workshop on Knowledge Discovery from Data Streams*, 2006.

[17] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognition Letters*, vol. 33, no. 2, pp. 191–198, 2012.

[18] R. Vlasveld, "Temporal Segmentation using Support Vector Machines in the context of Human Activity Recognition," Ph.D. dissertation, Utrecht University, 2014.

[19] K. Murao and T. Terada, "A recognition method for combined activities with accelerometers," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. ACM, 2014, pp. 787–796.

[20] H. Ombao, J. Heo, and D. Stoffer, "Statistical Analysis of Seismic Signals: An Almost Real-Time Approach," in *Time Series Analysis and Applications to Geophysical Systems*, D. Brillinger and E. Robinson, Eds. New York: Springer Verlag, 2004, pp. 53–72.

[21] T. Dasu, S. Krishnan, D. Lin, S. Venkatasubramanian, and K. Yi, "Change (detection) you can believe in: finding distributional shifts in data streams," in *Advances in Intelligent Data Analysis VIII*. Springer, 2009, pp. 21–34.

[22] Y. Kawahara, T. Yairi, and K. Machida, "Change-point detection in time-series data based on subspace identification," in *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*. IEEE, 2007, pp. 559–564.

[23] R. P. Adams and D. J. MacKay, "Bayesian Online Changepoint Detection," Cavendish Laboratory, Cambridge, Cambridge, UK, Tech. Rep., 2007.

[24] A. Bifet and R. Gavalda, "Learning from Time-Changing Data with Adaptive Windowing." in *SIAM International Conference on Data Mining*. Minneapolis, USA: SIAM, 2007, pp. 443–448.

[25] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Advances in neural information processing systems*, 2008, pp. 1433–1440.

[26] M. Yamada, A. Kimura, F. Naya, and H. Sawada, "Change-point detection with feature selection in high-dimensional time-series data," in *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. AAAI Press, 2013, pp. 1827–1833.

[27] A. C. Rencher, *Methods of Multivariate Analysis*, 2nd ed. New York, USA: John Wiley & Sons, 2002.

[28] C. E. Bonferroni, "Il Calcolo delle Assicurazioni su Gruppi di Teste," in *Studii in Onore del Profesor S. O. Carboni Roma*, 1936.

[29] S. R. Narum, "Beyond bonferroni: Less conservative analyses for conservation genetics," *Conservation Genetics*, vol. 7, no. 5, pp. 783–787, 2006.

[30] Shimmer. Shimmer Wearable Sensing Technology, Accessed October 2016. [Online]. Available: http://www.shimmersensing.com/

[31] J. M. Lucas and M. S. Saccucci, "Exponentially weighted moving average control schemes: properties and enhancements," *Technometrics*, vol. 32, no. 1, pp. 1–12, 1990.

[32] M. B. Khoo, "An extension for the univariate exponentially weighted moving average control chart," *Matematika*, vol. 20, pp. 43–48, 2004.

[33] J. Cannon, P. a. Krokhmal, Y. Chen, and R. Murphey, "Detection of temporal changes in psychophysiological data using statistical process control methods." *Computer methods and programs in biomedicine*, vol. 107, no. 3, pp. 367–81, Sep. 2012.

[34] T. Lovett and E. O'Neill, "Capturing transitions between users' semantically meaningful places using mobile devices," in *Proceedings of the 1st ACM workshop on Mobile systems for computational social science*. New York, New York, USA: ACM Press, 2012, pp. 11—-16.

[35] M. M. Salvador, B. Gabrys, and I. Žliobait, "Online Detection of Shutdown Periods in Chemical Plants: A Case Study," *Procedia Computer Science*, vol. 35, pp. 580–588, 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1877050914011041

[36] T. R. Agus, S. J. Thorpe, C. Suied, and D. Pressnitzer, "Characteristics of human voice processing," *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pp. 509–512, May 2010.

[37] "Neurophysiological correlates of age-related changes in working memory capacity." *Neuroscience letters*, vol. 392, no. 1-2, pp. 32–7, Jan. 2006.

[38] A. T. WELFORD, "The 'psychological refractory period' and the timing of high-speed performance – a review and a theory," *British Journal of Psychology. General Section*, vol. 43, no. 1, pp. 2–19, 1952.

[39] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, "A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer." *IEEE transactions on information technology in biomedicine : a publication of the IEEE Engineering in Medicine and Biology Society*, vol. 14, no. 5, pp. 1166–72, Sep. 2010.
[40] L. Bao and S. Intille, "Activity recognition from user-annotated acceleration data," in *Pervasive Computing*, A. Ferscha and F. Mattern, Eds. Linz/Vienna, Austria: Springer Berlin / Heidelberg, 2004, pp. 1–17.
[41] ComTech. Smartphone OS Sales Market Share Kantar Worldpanel ComTech, Accessed October 2016. [Online]. Available: http://www.kantarworldpanel.com/global/smartphone-os-market-share/
[42] Motorola. Motorola Moto G (2nd Gen), Accessed October 2016. [Online]. Available: http://www.motorola.com/we/products/moto-g-gen-2

Timothy Patterson Timothy received B.Sc. and Ph.D. degrees from Ulster University in 2009 and 2013 respectively. He is currently employed as a Research Associate within the Connected Health and Innovation Centre (Ulster University) where his current research interests lie in developing technological approaches to self-management of chronic conditions, activity change detection and ambient assistive living technologies.

Naveed Khan Naveed Khan received his B.Sc. degree in Computer Science in 2006 from Hazara University, Pakistan, and M.S degree in Computer Science in 2012 from King Saud University, Saudi Arabia. In September 2012, He joined the Department of Computer Science at King Saud University, Saudi Arabia as a Research Assistant and started working on Wireless Body Area Networks. Currently, He is doing his PhD degree in the field of Computer Science from the School of Computing and Information Engineering, Ulster University, Northern Ireland, UK. His current area of research is change detection in health sensor data.

Sally McClean Sally McClean received her first degree in Mathematics from Oxford University, and then obtained a MSc in Mathematical Statistics and Operational Research from Cardiff University, followed by a PhD on Markov and semi-Markov models at Ulster University. She is currently Professor of Mathematics at Ulster University. Her main research interests are in Stochastic Modelling and Optimisation, particularly for Healthcare Planning, and Computer Science, specifically Databases, Sensor Technology and Telecommunications. She has been grantholder on over 7M worth of funding, mainly from the EPSRC and other UK government sources. Sally is a Fellow of the Royal Statistical Society, Fellow of the Operational Research Society, past President of the Irish Statistical Association and Member of the IEEE. She has published over three hundred papers and was a recipient of Ulster Universitys Senior Distinguished Research Fellowship.

Chris Nugent Chris Nugent (S96A99M03) received the B.Eng. degree in electronic systems and the D.Phil. degree in Biomedical Engineering from the University of Ulster, Jordanstown, U.K., in 1995 and 1998, respectively. In 1998, he took the post of Research Fellow at the University of Ulster and now currently holds the post of Professor of Biomedical Engineering. His research interests include the design and evaluation of pervasive solutions in smart environments for ambient assisted living applications.

Shuai Zhang Shuai Zhang is a Lecturer in the School of Computing and Mathematics at the Ulster University. Her research interest is on Intelligent Data Analysis in the application areas of Connected Health and Ambient Assistive Living. Her current research includes activity and behavioural recognition in smart environment, change point detection for sensor data annotation and modelling user engagement with and adoption of assistive technologies for people with dementia.

Ian Cleland Ian Cleland is a Research Fellow in Computer Science within the Northern Ireland Connected Health Innovation Competence Centre (NI-CHIC), Ulster University. Ian received his B.Sc. in Biomedical Engineering in 2009 and a PhD in Computer Science in 2012, both from the Ulster University. his research interests include activity recognition, pervasive computing, the quantified self, self-management and technology facilitated behavior change.

Qin Ni is currently a PhD student in Telematics for Knowledge and Information Society research group (T¿SIC) at Universidad Politcnica de Madrid. She received her MS in Software and System from Universidad Politcnica de Madrid. Her research interests include pervasive computing, context-aware system, activity modelling and recognition, smart home application development.